(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平11－345127

(43)公開日　平成11年(1999)12月14日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/45		Ｇ０６Ｆ　9/44	３２２Ｈ

審査請求　有　　請求項の数９　ＯＬ　（全 16 頁）

(21)出願番号　特願平11－83570

(22)出願日　平成11年(1999)３月26日

(31)優先権主張番号　特願平10－88473

(32)優先日　平10(1998)４月１日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　中島　雅逸
大阪府門真市大字門真1006番地　松下電器産業株式会社内

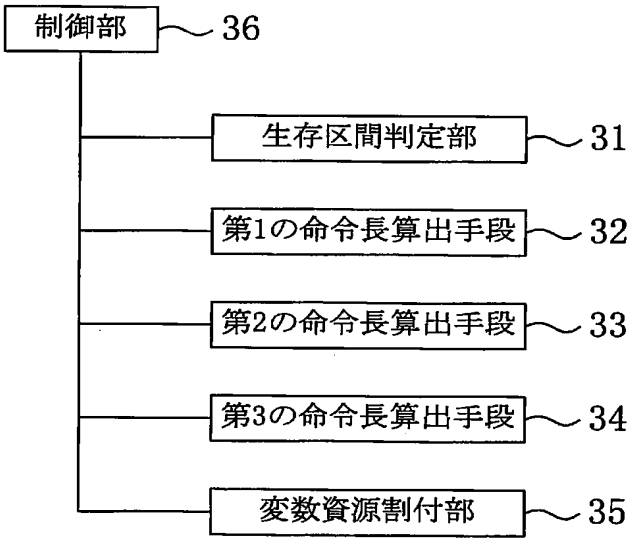
(74)代理人　弁理士　前田　弘　（外１名）

(54)【発明の名称】　コンパイラ

(57)【要約】　（修正有）

【課題】　命令コードサイズの小さい機械語プログラムを生成できるコンパイラを提供する。

【解決手段】　第１の命令列長算出手段３２は、ソースプログラム中の各変数を、第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けた場合の命令列長を算出する。第２の命令列長算出手段３３は、ソースプログラム中の各変数を、前記第１の命令フォーマットとは命令長が異なる第２の命令フォーマットで記述された命令を使用して、前記第１のレジスタ資源とは異なる第２のレジスタ資源に割り付けた場合の命令列長を算出する。前記算出結果に応じて資源割り付けを行う。



1

【特許請求の範囲】

【請求項１】　複数の命令からなるソースプログラムを入力し、このソースプログラムを機械語プログラムに翻訳するコンパイラであって、

前記ソースプログラム中の各変数を、第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けた場合の命令列の長さを算出する第１の命令列長算出手段と、

前記ソースプログラム中の各変数を、前記第１の命令フォーマットとは命令長が異なる第２の命令フォーマットで記述された命令を使用して、前記第１のレジスタ資源とは異なる第２のレジスタ資源に割り付けた場合の命令列の長さを算出する第２の命令列長算出手段とを備え、

前記第１及び第２の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とするコンパイラ。

【請求項２】　前記ソースプログラム中の各変数を、メモリに割り付けた場合の命令列の長さを算出する第３の命令列長算出手段を備え、

前記第１、第２及び第３の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とする請求項１記載のコンパイラ。

【請求項３】　前記第１及び第２の命令長算出手段が算出した命令長算出結果を比較して、命令列長が小さい資源割り付けを行うことを特徴とする請求項１記載のコンパイラ。

【請求項４】　前記第１のレジスタ資源は、前記第２のレジスタ資源の一部であることを特徴とする請求項１記載のコンパイラ。

【請求項５】　前記第１の命令フォーマットの命令長は、前記第２のフォーマット命令フォーマットの命令長より短いことを特徴とする請求項４記載のコンパイラ。

【請求項６】　前記ソースプログラム中の各変数の中で、使用頻度が高い変数を優先的に第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けることを特徴とする請求項１記載のコンパイラ。

【請求項７】　前記ソースプログラム中の各変数について、前記第１のレジスタ資源に割り付けられた変数を操作する場合には、前記第１の命令フォーマットを優先的に使用してその操作を記述することを特徴とする請求項６記載のコンパイラ。

【請求項８】　前記プログラム中の各変数の中で、使用頻度が高い変数、及びこの変数と共に使用される変数を、優先的に第１のレジスタ資源に割り付けることを特徴とする請求項６記載のコンパイラ。

【請求項９】　複数の命令からなるソースプログラムを翻訳した機械語プログラムを記録したコンピュータ読み取り可能な記録媒体であって、

前記機械語プログラムは、

2

第１のレジスタ資源を用いて第１の命令フォーマットで記述された命令と、

前記第１のレジスタ資源とは異なる第２のレジスタ資源を用いて、前記第１の命令フォーマットとは命令長が異なる第２の命令フォーマットで記述された命令とが混在し、前記第１の命令フォーマット及び第２の命令フォーマットは前記命令中の特定の領域の値により識別されることを特徴とする記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、高級プログラミング言語で書かれたソースプログラムを機械語プログラムに翻訳するコンパイラに関する。

【０００２】

【従来の技術】近年、Ｃ言語などの高級プログラミング言語でプログラムを記述し、プログラムの開発効率を高めることが盛んに行われている。高級プログラミング言語を用いることによって、プログラマは、プログラム中の数値の保持、演算、転送等の処理を、変数を用いた演算（ステップ）によって、任意に定義し、また、必要な個数だけ用いることができるため、プログラマは自由にプログラムを記述することができる。また記述されたプログラム（以下、ソースプログラムと称する）は、コンパイラによってコンパイルされることにより、コンピュータが理解可能な機械語プログラムとなる。この機械語プログラム中の演算は、機械語命令によって表現され、また、当該機械語命令は、レジスタ又はメモリをオペランドとするので、前記変数には、レジスタ又はメモリを割り付ける必要がある。この割り付け処理は、資源割り付け処理と称される。この資源割り付け処理が最適であれば、前記機械語プログラムのコードサイズを最小となる。

【０００３】一般的には、各変数に対して、レジスタを割り付けた場合の方が、メモリを割り付けた場合に比較して、コードサイズ的にも、実行時間的にも良好な結果が得られるが、レジスタは使用できる個数が少数であるため、如何に効率良くレジスタ資源の割り付けを行って、レジスタをオペランドとする機械語命令を使用するかということが、資源割り付け処理における最適化である。従来の資源割り付けの最適化方式としては、各々の変数が格納されている値が有効となる区間（変数の生存区間と称する。）に基づき、同一レジスタに割り付け変数はどれとどれであるかを検査し、その結果を用いて資源割り付けを行う方式が採用されている。

【０００４】一方、本願発明者は、特願平１０－５９６８０において、以下のような２種の命令フォーマット及びレジスタモデルを使用するデータ処理装置を提案している。図１０から図２０において、第１の命令フォーマットの概要を示す。

【０００５】第１の命令フォーマットは、最小命令語長

を１バイトとした可変長命令であり、レジスタアドレス指定フィールドとしては、２ビットのフィールドが使用される。従って、１つのレジスタアドレス指定フィールドで４個のレジスタが指定可能である。本アーキテクチャでは、４個のアドレスレジスタと、４個のデータレジスタが定義される。命令動作として、アドレスレジスタを使用するか、データレジスタを使用するかを区別することにより、命令で前記合計８個のレジスタを使用できるように構成される。

【０００６】図１０は、最小命令語長である１バイト目の第１の命令フィールドが、オペレーション指定フィールドと、任意の数のレジスタアドレス指定フィールドとから構成される第１の命令フォーマット（１）のビット割り付けを示す。

【０００７】第１の命令フォーマット（１）-(a) は、第１の命令フィールド内に２ビットのレジスタアドレス指定フィールドを２フィールド含み、最小命令語長である１バイトで構成される命令フォーマットであり、２つのオペランドが指定可能な命令フォーマットである。

【０００８】第１の命令フォーマット（１）-(b) は、第１の命令フィールド内に２ビットのレジスタアドレス指定フィールドを２フィールド含み、更に付加情報フィールドを追加して、合計２バイト以上の命令語長を持つ命令フォーマットである。

【０００９】第１の命令フォーマット（１）-(c) は、第１の命令フィールド内に２ビットのレジスタアドレス指定フィールドを１つ含み、最小命令語長である１バイトで構成される命令フォーマットであり、１つのオペランドが指定可能な命令フォーマットである。

【００１０】第１の命令フォーマット（１）-(d) は、第１の命令フィールド内に２ビットのレジスタアドレス指定フィールドを１フィールド含み、更に付加情報フィールドを追加した２バイト以上の命令語長を持つ命令フォーマットである。

【００１１】第１の命令フォーマット（１）-(e) は、第１の命令フィールド内にレジスタアドレス指定フィールドを含まず、最小命令語長である１バイトで構成される命令フォーマットであり、アドレスを用いたオペランド指定不可能な命令フォーマットである。

【００１２】第１の命令フォーマット（１）-(f) は、第１の命令フィールド内にレジスタアドレス指定フィールドを含まず、更に付加情報フィールドを追加した２バイト以上の命令語長を持つ命令フォーマットである。

【００１３】図１１は、図１０で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。

【００１４】図１２は、最小命令語長である１バイト目の第１の命令フィールドが命令語長指定フィールドからなり、第２の命令フィールドが、オペレーション指定フィールドと、任意の数のレジスタアドレス指定フィールドとから構成される第１の命令フォーマット（２）のビット割り付けを示したものである。

【００１５】第１の命令フォーマット（２）-(a) は、第２の命令フィールド内に２ビットのレジスタアドレス指定フィールドを２フィールド含み、２バイトで構成される命令フォーマットであり、２つのオペランドが指定可能な命令フォーマットである。

【００１６】第１の命令フォーマット（２）-(b) は、第２の命令フィールド内に２ビットのレジスタアドレス指定フィールドを２フィールド含み、更に付加情報フィールドを追加して、合計３バイト以上の命令語長を持つ命令フォーマットである。

【００１７】第１の命令フォーマット（２）-(c) は、第２の命令フィールド内に２ビットのレジスタアドレス指定フィールドを１つ含み、２バイトで構成される命令フォーマットであり、１つのオペランドが指定可能な命令フォーマットである。

【００１８】第１の命令フォーマット（２）-(d) は、第２の命令フィールド内に２ビットのレジスタアドレス指定フィールドを１フィールド含み、更に付加情報フィールドを追加した３バイト以上の命令語長を持つ命令フォーマットである。

【００１９】第１の命令フォーマット（２）-(e) は、第２の命令フィールド内にレジスタアドレス指定フィールドを含まず、２バイトで構成される命令フォーマットであり、アドレスを用いたオペランド指定が不可能な命令フォーマットである。

【００２０】第１の命令フォーマット（２）-(f) は、第２の命令フィールド内にレジスタアドレス指定フィールド含まず、更に、付加情報フィールドを追加した２バイト以上の命令語長を持つ命令フォーマットである。

【００２１】図１３は、図１２で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。

【００２２】従って、前記図１０から図１３に示した第１の命令フォーマットは、第１の命令フィールドを基本命令語長とし、第１から第Mの命令フィールドを最長命令語長Mとして、N命令語長（Nは１～M間での整数）の可変長命令を特定するものであって、最小命令語長が１バイトであるという他にない特徴を備え、プログラムサイズの縮小に適した命令フォーマットとなっている。

【００２３】図１４は、本データ処理装置に備える第１のレジスタファイル２２０を示す。この第１のレジスタファイル２２０は、４個のアドレスレジスタＡ０～Ａ３と、４個のデータレジスタＤ０～Ｄ３と、スタックポインタＳＰ２２３と、内部のステータス情報及び制御情報を保持するPSW(Processor Status Word)２２４と、プログラムカウンタＰＣ２２５とを含む。

5

【００２４】更に、図１５は、前記第１のレジスタファイル２２０のアドレスレジスタＡ０～Ａ３及びデータレジスタＤ０～Ｄ３へのアクセスについて、より詳細に示した図である。同図は、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コードでのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及びアクセスする対象となる物理的なレジスタ名を一覧にして示したものである。

【００２５】図１５に示すように、第１の命令フォーマットにおいては、４個のアドレスレジスタＡ０～Ａ３へアクセスするために命令内に指定される命令アドレス指定フィールドと、４個のデータレジスタＤ０～Ｄ３へアクセスするために命令内に指定される命令アドレス指定フィールドとは、全く同一である。即ち、レジスタのアドレスを指定するために２ビット命令アドレス指定フィールドを使用し、命令動作そのものでアドレスレジスタにアクセスするか、データレジスタにアクセスするかを区別させている。

【００２６】次に、本アーキテクチャの基本命令フォーマットである前記図１０及び図１２の第１の命令フォーマットに対して、追加拡張する第２の命令フォーマットのビット割り付けを図１６に示す。

【００２７】図１６に示した第２の命令フォーマットのビット割り付けは、最小命令語長である１バイト目の第１の命令フィールドが命令語長指定フィールドからなり、第２及び第３の命令フィールドが、オペレーション指定フィールドと、任意の数のレジスタアドレス指定フィールドとから構成される。第２の命令フォーマットにおけるレジスタアドレス指定フィールドは４ビットから構成される。

【００２８】図１６において、第２の命令フォーマット（a）は、第３の命令フィールド内に４ビットのレジスタアドレス指定フィールドを２フィールド含み、３バイトで構成される命令フォーマットであり、２つのオペランドが指定可能な命令フォーマットである。

【００２９】第２の命令フォーマット（b）は、第３の命令フィールド内に４ビットのレジスタアドレス指定フィールドを２フィールド含み、更に、付加情報フィールドを追加して、合計４バイト以上の命令語長を持つ命令フォーマットである。

【００３０】第２の命令フォーマット（c）は、第３の命令フィールド内に４ビットのレジスタアドレス指定フィールドを１つ含み、３バイトで構成される命令フォーマットであり、１つのオペランドが指定可能な命令フォーマットである。

【００３１】第２の命令フォーマット（d）は、第３の命令フィールド内に４ビットのレジスタアドレス指定フィールドを１フィールド含み、更に、付加情報フィールドを追加した４バイト以上の命令語長を持つ命令フォーマットである。

6

【００３２】従って、前記第２の命令フォーマットも、第１の命令フィールドを基本命令語長とし、第１から第Ｍの命令フィールドを最長命令語長Ｍとして、Ｎ命令語長（Ｎは１～Ｍ間での整数）の可変長命令を特定する。

【００３３】図１７は、図１６で示された個々のビット割り付けについて、具体的な命令のリストの一部を示したものである。左側に命令のニーモニックを、右側に命令の動作を各々示している。ニーモニックの中で、Rm、Rn、又はRiは、レジスタアドレスの指定を表すが、指定できるレジスタとして、図１８に示されるような第２のレジスタファイル１２０が定義され、４個のアドレスレジスタＡ０～Ａ３、４個のデータレジスタＤ０～Ｄ３、及び８個の拡張レジスタＥ０～Ｅ７から構成される１６個の汎用レジスタである。更に、この第２のレジスタファイル１２０は、スタックポインタSP１２２と、内部のステータス情報及び制御情報を保持するPSW(Processor Status Word)１２３と、プログラムカウンタPC１２４とを含んでいる。

【００３４】図１９は、第１の命令フォーマットで定義された命令を実行する際に、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コード上でのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及び、アクセスする対象となる物理的なレジスタ名を一覧にして示したものである。第１の命令フォーマットでは、レジスタ指定フィールドは２ビットしかないが、汎用レジスタは１６個で４ビットのアドレスでアクセスする必要がある関係から、アドレスの変換をする必要がある。例えば、アドレスレジスタＡ０をアクセスする際には、物理的なアドレス番号として“１０００”が、データレジスタＤ１をアクセスする際には、物理的なアドレス番号として、“１１０１”を生成し、汎用レジスタファイル１２１に出力する必要がある。

【００３５】図２０は、第２の命令フォーマットで定義された命令を実行する際に、命令の中で指定されるレジスタ名と、レジスタアドレス指定フィールドで指定される命令コード上でのビット割り付けと、物理的なレジスタにアクセスするための物理的なレジスタ番号、及び、アクセスする対象となる物理的なレジスタ名を一覧にして示したものである。第２の命令フォーマットでは、４ビットのレジスタアドレス指定フィールドを有しているので、その４ビットをそのまま物理的なレジスタ番号として指定することになる。

【００３６】

【発明が解決しようとする課題】本願発明者が特願平１０－５９６８０において提案したようなデータ処理装置においては、従来のコンパイラが行っているように単にレジスタをメモリよりも優先して資源割り付けを行うと、以下のような問題が存在する。

【００３７】１）第１のレジスタファイル（レジスタ資源）に割り付けられた変数を第１の命令フォーマットで記述された命令を使用して処理する場合と、第２のレジスタファイルに割り付けられた変数を第２の命令フォーマットで記述された命令を使用して処理する場合とで命令長が異なるため、両者の何れをも優先的に扱わず均一に扱うことでは、命令コードサイズは最小にならない。即ち、従来のコンパイラでは、各変数を第１のレジスタファイルに優先的に割り付けるか、又は第２のレジスタファイルに優先的に割り付けるかについて考慮していない。例えば、各変数をその出現順に第２のレジスタファイルに割り付け、これ等変数を第２の命令フォーマットで記述された命令を使用して処理する場合には、第２の命令フォーマットの命令長が第１の命令フォーマットの命令長よりも長い以上、命令コードサイズは大きくなる。

【００３８】２）第２の命令フォーマットで記述された命令を使用して変数を処理する場合に対して、データをメモリからレジスタに転送するデータ転送命令を含む命令列では、第１の命令フォーマットを使用して変数を処理した方が、命令数は増加しても命令長は短くなる場合がある。従って、単純に、変数をメモリよりもレジスタ資源に優先的に割り付けることでは、コードサイズは最小にならない。

【００３９】本発明の目的は、命令フォーマットによって扱えるレジスタ資源が異なり、且つ使用する命令フォーマットに応じて命令長が異なる命令セットを有するプロセッサにおいて、命令コードサイズが小さい機械語プログラムを生成するコンパイラを提供することにある。

【００４０】

【課題を解決するための手段】前記課題を解決するために、本発明では、頻繁に使用される変数が、命令長の短いフォーマットで使用可能なレジスタに割り付けられ、且つそれ等変数が前記命令長の短いフォーマットを使用して処理される機械語プログラムを得るようにする。

【００４１】すなわち、請求項１記載の発明のコンパイラは、複数の命令からなるソースプログラムを入力し、このソースプログラムを機械語プログラムに翻訳するコンパイラであって、前記ソースプログラム中の各変数を、第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けた場合の命令列の長さを算出する第１の命令列長算出手段と、前記ソースプログラム中の各変数を、前記第１の命令フォーマットとは命令長が異なる第２の命令フォーマットで記述された命令を使用して、前記第１のレジスタ資源とは異なる第２のレジスタ資源に割り付けた場合の命令列の長さを算出する第２の命令列長算出手段とを備え、前記第１及び第２の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とする。

【００４２】請求項２記載の発明は、前記請求項１記載のコンパイラにおいて、前記ソースプログラム中の各変数を、メモリに割り付けた場合の命令列の長さを算出する第３の命令列長算出手段を備え、前記第１、第２及び第３の命令長算出手段が算出した命令長算出結果を用いて資源割り付けを行うことを特徴とする。

【００４３】請求項３記載の発明は、前記請求項１記載のコンパイラにおいて、前記第１及び第２の命令長算出手段が算出した命令長算出結果を比較して、命令列長が小さい資源割り付けを行うことを特徴とする。

【００４４】請求項４記載の発明は、前記請求項１記載のコンパイラにおいて、前記第１のレジスタ資源は、前記第２のレジスタ資源の一部であることを特徴とする。

【００４５】請求項５記載の発明は、前記請求項４記載のコンパイラにおいて、前記第１の命令フォーマットの命令長は、前記第２のフォーマット命令フォーマットの命令長より短いことを特徴とする。

【００４６】請求項６記載の発明は、前記請求項１記載のコンパイラにおいて、前記ソースプログラム中の各変数の中で、使用頻度が高い変数を優先的に第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けることを特徴とする。

【００４７】請求項７記載の発明は、前記請求項６記載のコンパイラにおいて、前記ソースプログラム中の各変数について、前記第１のレジスタ資源に割り付けられた変数を操作する場合には、前記第１の命令フォーマットを優先的に使用してその操作を記述することを特徴とする。

【００４８】請求項８記載の発明は、前記請求項６記載のコンパイラにおいて、前記プログラム中の各変数の中で、使用頻度が高い変数、及びこの変数と共に使用される変数を、優先的に第１のレジスタ資源に割り付けることを特徴とする。

【００４９】請求項９記載の発明の記録媒体は、複数の命令からなるソースプログラムを翻訳した機械語プログラムを記録したコンピュータ読み取り可能な記録媒体であって、前記機械語プログラムは、第１のレジスタ資源を用いて第１の命令フォーマットで記述された命令と、前記第１のレジスタ資源とは異なる第２のレジスタ資源を用いて、前記第１の命令フォーマットとは命令長が異なる第２の命令フォーマットで記述された命令とが混在し、前記第１の命令フォーマット及び第２の命令フォーマットは前記命令中の特定の領域の値により識別されることを特徴とする。

【００５０】以上の構成から、本発明では、命令フォーマットの種類に応じて扱えるレジスタ資源が異なり、且つ命令フォーマットの種類に応じて命令長が異なる命令セットを有するプロセッサにおいて、各変数を第１の命令フォーマットで記述された命令を使用して第１のレジスタ資源に割り付けた場合の命令列長の長さと、各変数を第２の命令フォーマットで記述された命令を使用して

第２のレジスタ資源に割り付けた場合の命令列長の長さとが算出され、その算出結果に応じて適切な資源割り付けが行われるので、コードサイズの小さい機械語プログラムが生成される。

【００５１】

【発明の実施の形態】以下、本発明の実施の形態について、図１から図５を用いて説明する。

【００５２】最初に本発明におけるコンパイラが対象とするプロセッサの機械語命令セットについてであるが、命令のフォーマットによって扱えるレジスタ資源が異なり、且つ、命令のフォーマットに応じて命令長が異なるような機械語命令セットを有するプロセッサとして、特願平１０－５９６８０において提案した２種の命令フォーマットを有するデータ処理装置とする。

【００５３】図１は、本発明の実施の形態におけるコンパイラの構成図である。コンパイラは、構文解析手段１と、最適化手段２と、資源割り付け手段３と、コード生成手段４とで構成されている。

【００５４】構文解析手段１は、ファイルとして記憶されているソースプログラム５の字句解析、構文解析、及び意味解析を行う。解析結果は、中間言語プログラムとして出力される。

【００５５】最適化手段２は、最終的に生成される機械語プログラム６のプログラムサイズを削減し、実行時間を短縮させる目的で中間言語プログラムの最適化を行う。

【００５６】資源割り付け手段３は、プログラムの変数の生存区間を求め、生存区間毎に各変数に対して資源であるレジスタやメモリを割り付けると共に、同一動作に対しての最適な命令の割り付けも行う。

【００５７】コード生成手段４は、最適化された中間言語プログラムを、資源割り付け手段３の割り付け結果に従って、ターゲットマシンの機械語命令に変換し、機械語プログラム６として出力する。

【００５８】尚、構文解析手段１、最適化手段２、コード生成手段４については公知であるので、詳細な説明を省略する。

【００５９】図２は、本発明の実施の形態におけるコンパイラの資源割り付け手段３の構成図である。同図において、３１は各変数に対しての生存区間を判定する生存区間判定部である。３２は変数を第１の命令フォーマットを持つ命令群を用いてレジスタに割り付けた場合の命令列長を算出する第１の命令長算出手段である。第１の命令フォーマットでは、一部のレジスタしか扱うことができない。３３は変数を第２の命令フォーマットを持つ命令群を用いてレジスタに割り付けた場合の命令長を算出する第２の命令長算出手段である。第２の命令フォーマットでは全てのレジスタ資源を扱うことができる。３４は変数をメモリに割り付けた場合の命令長を算出する第３の命令長算出手段である。３５は生存区間判定部３１の生存区間の判定結果と、第１から第３の命令長算出手段３２～３４に応じて、変数資源の割り付けを行う変数資源割り付け部である。３６は資源割り付け手段３の制御部３６を除く全体を制御する制御部である。

【００６０】以上のように構成されたコンパイラの資源割り付けについて、以下、図面を用いて、その動作を述べる。図３は、本発明の実施の形態におけるコンパイラの制御部３６の資源割り付けの処理フローを示したフローチャートである。

【００６１】ここで、コンパイラの資源割付けを行う対象とする中間コードプログラム例を図４に示す。図４に示された中間コードｓ１～ｓ８を今回の資源割付けの対象となる基本ブロックとする。説明を簡単にするために、ここでは、基本ブロック内の資源割り付けについて説明するが、基本ブロックを越えての資源割付けにおいても同様に説明できる。

【００６２】また、説明を簡単にするために、レジスタ資源の割り付けに関しては、以下の条件で行うものとする。

【００６３】１）第１の命令フォーマットで操作できるレジスタ資源（Ａ０～Ａ３，Ｄ０～Ｄ３）の中で、アドレスレジスタＡ０～Ａ３については、ポインタとして使用するものとし、データ変数を格納しないものとする。

【００６４】２）第１の命令フォーマットで操作できるレジスタ資源（Ａ０～Ａ３，Ｄ０～Ｄ３）の中で、データレジスタＤ０～Ｄ３は、全てデータ変数の格納及び操作に使用できるが、そのうち、レジスタＤ０、Ｄ１についてはワークレジスタとして使用し、変数を割り付けないものとする。

【００６５】３）第２の命令フォーマットで操作できるレジスタ資源（Ａ０～Ａ３，Ｄ０～Ｄ３、Ｅ０～Ｅ７）の中で、拡張レジスタＥ０～Ｅ７については、データでもアドレスでも自由に使用できるものとする。

【００６６】これ等の条件は、説明を簡単にするために定めたものであって、本発明に何らの制約を与えるものではない。

【００６７】このフローチャートに従って、図４における中間コードプログラムについて資源割り付けを行う場合の処理について説明する。

【００６８】先ず、ステップ３０１では、基本ブロック内部の全ての変数について、個々の変数の生存区間や参照頻度を調べる。ここで、レジスタ割付けの対象となるのは、第１の命令フォーマットで操作できるレジスタ資源のみ、具体的には、レジスタＡ０～Ａ３、Ｄ０～Ｄ３のみである。その結果によって、変数資源割付部３５が第１の命令フォーマットで操作できるレジスタ資源に対して、レジスタ割付け可能かどうかを検査する。ここでは、どのような手法を用いてこれ等のレジスタに変数を割り付けるかということが重要ではなく、同一のオペレーションに対して第１の命令フォーマットの命令長の方

が第２の命令フォーマットの命令長に比べて小さいので、第１の命令フォーマットで扱えるレジスタ資源のみを対象にしたレジスタ資源割り付けを行うということが最も重要である。

【００６９】次に、ステップ３０２では、ステップ３０１の検査結果をもとに、第１の命令フォーマットで操作できるレジスタ資源のみで全ての変数が資源割付け可能かどうかを判断する。

【００７０】可能な場合は、第１の命令フォーマットで操作できるレジスタ資源のみに変数を割り付けて、資源割付けを終了する（ステップ３０３）。

【００７１】通常の基本ブロック内には、多くの変数が存在するので、ほとんどの場合は、第１の命令フォーマットで操作できるレジスタ資源に割り付けられない変数が存在する。ステップ３０２では、第１の命令フォーマットで操作できるレジスタ資源で全ての変数が割り付けられない場合、先ず、第１の命令フォーマットで操作できるレジスタ資源を優先的に割り付けを行う。この優先的な割り付け処理を以下に説明する。

【００７２】図５は、ステップ３０１で、第１の命令フォーマットで操作できるレジスタ資源に割り付ける変数を決定するために調べた、基本ブロック内での変数の生存区間と、参照頻度とを示した図である。この結果、最も参照頻度の高い変数であるｐ１とｐ２を第１の命令フォーマットで操作できるレジスタの内、レジスタＤ２とＤ３に割り付けるものとする。ここで、どの変数をレジスタに割り付けるかどうかの決定は、扱えるレジスタ資源の数と、各変数の生存区間、参照頻度によって決定されたものである。この場合、既述した条件によりレジスタＤ０及びＤ１は使用できないので、使用できるレジスタ資源はレジスタＤ２とＤ３との２つであり、最も使用頻度の高い２つの変数を割り付けている。

【００７３】次に、ステップ３０４では、変数資源割付部３５が第１の命令フォーマットで操作できるレジスタ資源に割り付けできなかった残りの変数について、第１の命令フォーマットでは操作できず、第２の命令フォーマットによってのみ操作できるレジスタ資源、具体的には、拡張レジスタＥ０～Ｅ７に各変数を割り付けるか、メモリ資源に割り付けるかの第１の組み合わせ（資源の割り付け）について場合分けを行う。

【００７４】更に、ステップ３０４では、制御部３６が第１の組み合わせの中でレジスタ資源に割り付けた場合には、ワークレジスタＤ０、Ｄ１を利用して、第１の命令フォーマットを使用するのが適当か、又はワークレジスタＤ０、Ｄ１を利用せずに第２の命令フォーマットを使用するのが適当かも含めて、第２の組み合わせ（命令の割り付け）を考える。拡張レジスタＥ０～Ｅ７及びレジスタＤ２、Ｄ３を使用した特定の第１の組み合わせに対する第２の組み合わせの例を図７～図９に示す。これ等の図面の説明は後に詳述する。最終的には、全ての組

み合わせについて、命令列長を算出するので、ここで、どの組み合わせを選択するかは問題にならない。

【００７５】以下に説明するステップ３０５～ステップ３０７では、前記第２の組み合わせの各々（図７～図９）に対して命令列長の算出を行う。

【００７６】つまり、ステップ３０５では、第１の命令フォーマットを使用してレジスタに資源割り付けした命令列長を算出する。ステップ３０６では、第２の命令フォーマットを使用してレジスタに資源割り付けした命令列長を算出する。

【００７７】同様に、ステップ３０７では、レジスタ資源ではなく、メモリ資源に変数を割り付け、第１の命令フォーマットを割り付けた場合の命令列長を算出する。ここでは、説明を簡単にするために、メモリ資源に変数を割り付けた場合のアドレスは、１６ビットの絶対アドレスで指定できるものとし、その変数に対してのメモリ／レジスタ間移動命令は、３バイトとする。またステップ３０７では、レジスタ資源ではなく、メモリ資源に変数を割り付け、第２の命令フォーマットを割り付けた場合の命令列長を算出する。ここでは、説明を簡単にするために、メモリ資源に変数を割り付けた場合のアドレスは、１６ビットの絶対アドレスで指定できるものとし、その変数に対してのメモリ／レジスタ間移動命令は、４バイトとする。

【００７８】ステップ３０８では、前記ステップ３０５～ステップ３０７の結果を用いて、この組み合わせにおける基本ブロック全体の命令列長を算出する。

【００７９】前記処理を全ての組み合わせにおいて実行し（ステップ３０９）、ステップ３１０では、その結果、基本ブロック内の命令列長が最小となる組み合わせを選択する）。

【００８０】以上のような処理で機械語プログラムサイズを最小化するコンパイラの処理について、図を用いて更に詳細に説明する。

【００８１】先ず、図６に、各変数のレジスタ割付けについて第１の命令フォーマットによってアクセス可能なレジスタ資源と、それ以外のレジスタ資源について、優先度を付けずに出現順にレジスタ資源への割り付けを行った場合（従来の方法）の変数の資源割付けと、その割り付けに対応して命令を割り付けた場合の機械語命令プログラムを示す。この場合の基本ブロックに対する命令列長は４６バイトになる。

【００８２】次に、図７に、図５で示された変数の生存区間と出現頻度をもとに、図３のステップ３０２の処理により、第１の命令フォーマットで操作可能なレジスタのみ（この場合、具体的には、Ｄ２レジスタとＤ３レジスタ）に、優先的にレジスタ割付けを行い、その結果、変数ｐ１をレジスタＤ２に、変数ｐ２をレジスタＤ３に割り付けた場合の変数の資源割り付けと、その割り付けに対応して命令を割り付けた場合の機械語命令プログラ

ムを示す。この場合の基本ブロックに対する命令列長は４２バイトとなり、図６で示した場合に比べて、４バイトのコードサイズの削減が図れていることが判る。

【００８３】更に、図８に、図７で説明した場合とレジスタ資源の割り付け方は同様であるが、命令フォーマットとして第２の命令フォーマットを選択する代りに第１の命令フォーマットを選択して、命令の割り付け方を変更した場合の機械語命令プログラムを示す。この場合、基本命令ブロックに対するトータルの命令長自体は、図７の場合より２バイト増加して、４４バイトになっているが、個々の中間言語毎に機械語命令列長を比較していくと、中間言語ｓ１及び中間言語ｓ８に関しては、これ等言語ｓ１、ｓ８が第１の命令フォーマットで記述された命令を含んで、各々、１バイトずつ機械語命令長が減少していることが判る。ここで、中間言語ｓ１では、使用頻度の高い変数ｐ１及びｐ２が第１のレジスタファイル２２０内のレジスタＤ２、Ｄ３に各々割り付けられる。また、これ等変数ｐ１、ｐ２と共に使用される変数ｔ１も第１のレジスタファイル２２０内のワークレジスタＤ０に一時的に割り付けられる。これにより、命令mov D2,D0 及び命令add D3,D0は、各々、第１の命令フォーマットを使用して１バイトで記述される。

【００８４】最後に、図７及び図８で示した結果をもとに、レジスタ資源の割り付け、及び命令の割り付けを最適化した機械語命令プログラムを図９に示す。図９では、図７及び図８の各中間言語の中で機械語命令長が短い方の中間言語、即ち、中間言語ｓ２～ｓ７では図７の中間言語ｓ２～ｓ７が、中間言語ｓ１、ｓ８では図８の中間言語ｓ１、ｓ８が採用される。尚、図７の中間言語ｓ２、ｓ３と図８の中間言語ｓ２、ｓ３とは機械語命令長が同一であるが、図７の中間言語ｓ２、ｓ３、即ち命令数が少なくて高速処理の可能な方の中間言語が採用される。その結果、図９では、図７の状態から、更に２バイトのコード削減が実現されており、図３及び図４で示した中間言語プログラムに対して、最もコードサイズの小さい機械語命令プログラムが生成されている。

【００８５】前記図９の機械語命令プログラムは、中間言語ｓ１の命令mov D2,D0 、命令add D3,D0 、及び中間言語ｓ８の命令add 5,D0が第１のレジスタファイル２２０内のレジスタＤ０、Ｄ２、Ｄ３を用いて第１の命令フォーマットで記述され、他の中間言語ｓ２～ｓ７が第２のレジスタファイル１２０を用いて第２の命令フォーマットで記述されていて、この機械語命令プログラムは、コンピュータ読み取り可能な記録媒体に記録される。

【００８６】尚、以上の具体例では、基本ブロックの各変数ｐ１、ｐ２、ｔ１～ｔ８をレジスタＤ２、Ｄ３、Ｅ０～Ｅ７に割り付けたが、資源割り付けが必要な変数が割り付け可能なレジスタの本数を越えて同時に存在する場合には、これ等変数は前記各レジスタとメモリ資源とに割り付けられる。この場合は、図３のステップＳ３０７においてメモリ資源に割り付けた場合の命令列長が算出される。このように変数の一部をメモリ資源に割り付ける場合、データ転送命令を含む命令列では、第１のレジスタファイル２２０内の何れかのレジスタを一時的に使用して第１の命令フォーマットが使用される。これにより、命令数は増加するが、命令列長は短くなり、命令コードサイズは小さくなる場合がある。

【００８７】

【発明の効果】以上説明したように、本発明のコンパイラによれば、命令フォーマットの種類に応じて扱えるレジスタ資源が異なり、且つ命令フォーマットの種類に応じて命令長が異なる命令セットを有するプロセッサにおいて、コードサイズの小さい機械語プログラムを生成できるコンパイラを提供できるという顕著な効果が得られる。

【図面の簡単な説明】

【図１】本発明の一実施の形態によるコンパイラの構成を示す構成図である。

【図２】同実施形態によるコンパイラの資源割り付け手段の構成を示す構成図である。

【図３】同実施形態によるコンパイラの資源割り付け手段における資源割り付けに関る処理フローを説明するフローチャート図である。

【図４】中間言語プログラムの一例を示す図である。

【図５】中間言語プログラム中の各変数の生存区間と参照頻度を示した図である。

【図６】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図７】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図８】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図９】変数の資源割付けとそれに対応した機械語命令プログラムの一例を示す図である。

【図１０】本発明によるコンパイラが対象とするデータ処理装置の第１の命令フォーマット（１）を示す図である。

【図１１】同データ処理装置の第１の命令フォーマット（１）の命令のリストの一部を示す図である。

【図１２】同データ処理装置の第１の命令フォーマット（２）を示す図である。

【図１３】同データ処理装置の第１の命令フォーマット（２）の命令のリストの一部を示す図である。

【図１４】同データ処理装置において、第１のレジスタファイルの構成を示すブロック図である。

【図１５】同データ処理装置の第１の命令フォーマットの命令の実行時のレジスタファイルのレジスタ番号を示す図である。

【図１６】同データ処理装置の第２の命令フォーマットを示す図である。

15

【図１７】同データ処理装置の第２の命令フォーマットの命令のリストの一部を示す図である。

【図１８】同データ処理装置のレジスタファイルの構成を示すブロック図である。

【図１９】同データ処理装置において、第１の命令フォーマットの命令実行時のレジスタファイルのレジスタ番号を示す図である。

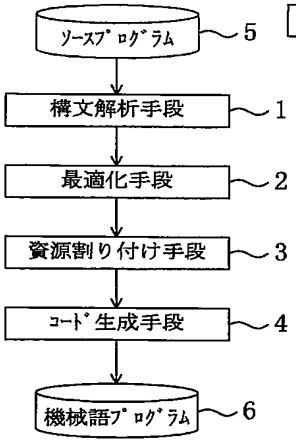
【図２０】同データ処理装置において、第２の命令フォーマットの命令の実行時のレジスタファイルのレジスタ番号を示す図である。

【符号の説明】

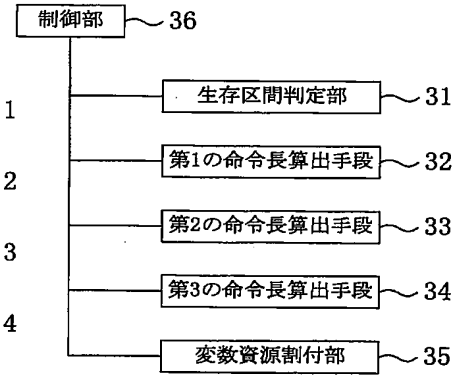
16

1　構文解析手段
2　最適化手段
3　資源割り付け手段
4　コード生成手段
３１　生存区間判定部
３２　第１の命令長算出手段
３３　第２の命令長算出手段
３４　第３の命令長算出手段
３５　変数資源割り付け部
３６　制御部

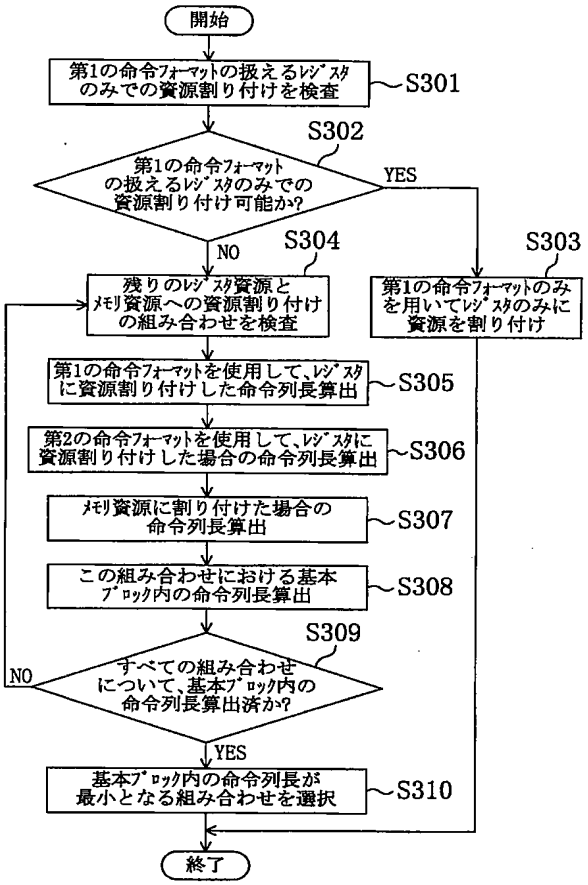
【図１】



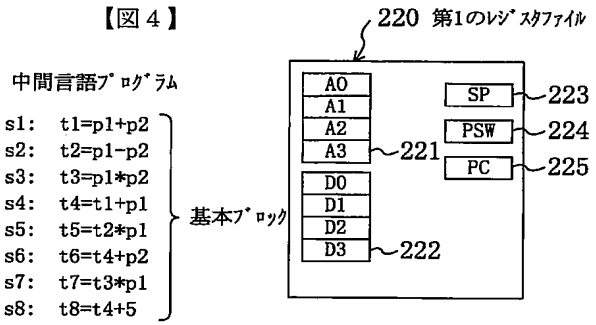
【図２】



【図３】

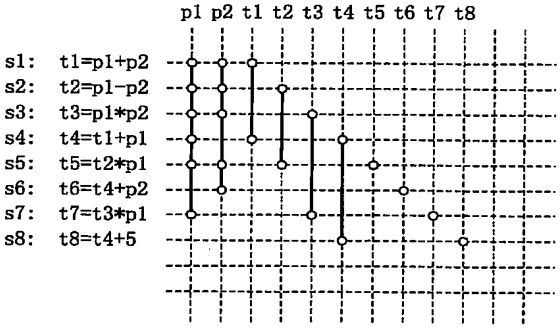


【図４】



【図１４】

【図５】



【図１５】

レジスタ名	命令コード上でのビット割り付け	物理的なレジスタ番号	物理的なレジスタ名
A0	00	00	アドレスレジスタ
A1	01	01	アドレスレジスタ
A2	02	02	アドレスレジスタ
A3	03	03	アドレスレジスタ
D0	00	00	データレジスタ
D1	01	01	データレジスタ
D2	02	02	データレジスタ
D3	03	03	データレジスタ

【図６】

各変数の資源割り付け

p1	E0
p2	E1
t1	E2
t2	E3
t3	E4
t4	E5
t5	E6
t6	E7
t7	D2
t8	D3

中間言語ﾌﾟﾛｸﾞﾗﾑ

```
s1:  t1=p1+p2
s2:  t2=p1-p2
s3:  t3=p1*p2
s4:  t4=t1+p1
s5:  t5=t2*p1
s6:  t6=t4+p2
s7:  t7=t3*p1
s8:  t8=t4+5
```

機械語命令ﾌﾟﾛｸﾞﾗﾑ

```
s1:  mov   E0, E2   3
     add   E1, E2   3

s2:  mov   E0, E3   3
     sub   E1, E3   3

s3:  mov   E0, E4   3
     mul   E1, E4   3

s4:  mov   E0, E5   3
     add   E2, E5   3

s5:  mov   E0, E6   3
     mul   E3, E6   3

s6:  mov   E1, E7   3
     add   E5, E7   3

s7:  mov   E0, D2   2
     mul   E4, D2   3

s8:  mov   E3, D3   2
     add    5, D3   3
                   ---
                   46byte
```

【図７】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

中間言語ﾌﾟﾛｸﾞﾗﾑ

```
s1:  t1=p1+p2
s2:  t2=p1-p2
s3:  t3=p1*p2
s4:  t4=t1+p1
s5:  t5=t2*p1
s6:  t6=t4+p2
s7:  t7=t3*p1
s8:  t8=t4+5
```

機械語命令ﾌﾟﾛｸﾞﾗﾑ

```
s1:  mov   D2, E0   2
     add   D3, E0   3

s2:  mov   D2, E1   2
     sub   D3, E1   3

s3:  mov   D2, E2   2
     mul   D3, E2   3

s4:  mov   D2, E3   2
     add   E0, D3   3

s5:  mov   D2, E4   2
     mul   E1, E4   3

s6:  mov   D3, E5   2
     add   E3, E5   3

s7:  mov   D2, E6   2
     mul   E2, E6   3

s8:  mov   E3, E7   3
     add    5, E7   4
                   ---
                   42byte
```

【図８】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

中間言語ﾌﾟﾛｸﾞﾗﾑ

```
s1:  t1=p1+p2
s2:  t2=p1-p2
s3:  t3=p1*p2
s4:  t4=t1+p1
s5:  t5=t2*p1
s6:  t6=t4+p2
s7:  t7=t3*p1
s8:  t8=t4+5
```

機械語命令ﾌﾟﾛｸﾞﾗﾑ

```
s1:  mov   D2, D0   1
     add   D3, D0   1
     mov   D0, E0   2

s2:  mov   D2, D0   1
     sub   D3, D0   2
     mov   D0, E1   2

s3:  mov   D2, D0   1
     mul   D3, D0   2
     mov   D0, E2   2

s4:  mov   D2, D0   1
     add   E0, D0   3
     mov   D0, E3   2

s5:  mov   D2, D0   1
     mul   E1, D0   3
     mov   D0, E4   2

s6:  mov   D3, D0   1
     add   E3, D0   3
     mov   D0, E5   2

s7:  mov   D2, D0   1
     mul   E2, D0   3
     mov   D0, E6   2

s8:  mov   E3, D0   2
     add    5, D0   2
     mov   D0, E7   2
                   ---
                   44byte
```

【図９】

各変数の資源割り付け

p1	D2
p2	D3
t1	E0
t2	E1
t3	E2
t4	E3
t5	E4
t6	E5
t7	E6
t8	E7

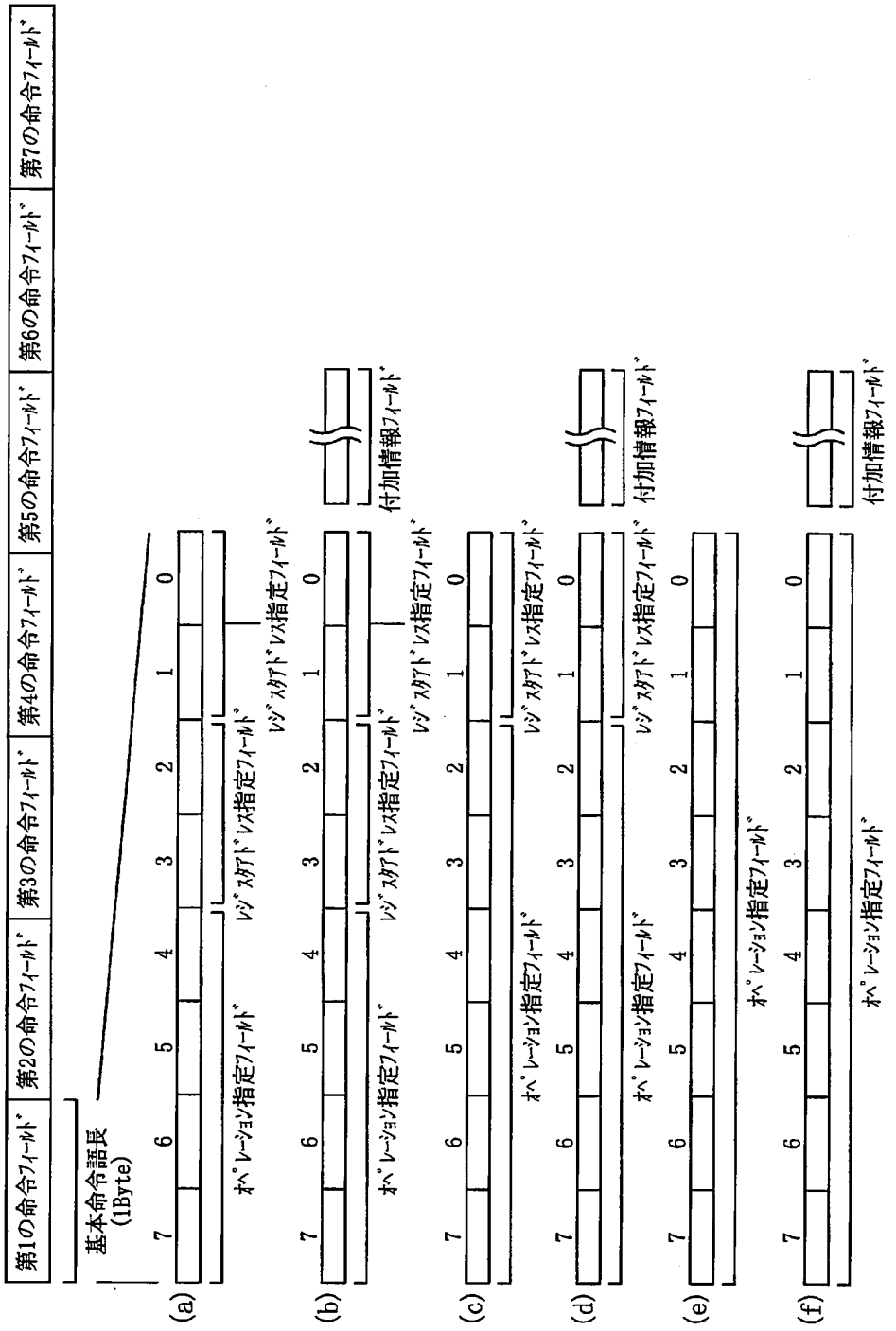
中間言語ﾌﾟﾛｸﾞﾗﾑ

```
s1:  t1=p1+p2
s2:  t2=p1-p2
s3:  t3=p1*p2
s4:  t4=t1+p1
s5:  t5=t2*p1
s6:  t6=t4+p2
s7:  t7=t3*p1
s8:  t8=t4+5
```

機械語命令ﾌﾟﾛｸﾞﾗﾑ

```
s1:  mov   D2, D0   1
     add   D3, D0   1
     mov   D0, E0   2

s2:  mov   D2, E1   2
     sub   D3, E1   3

s3:  mov   D2, E2   2
     mul   D3, E2   3

s4:  mov   D2, E3   2
     add   E0, E3   3

s5:  mov   D2, E4   2
     mul   E1, E4   3

s6:  mov   D3, E5   2
     add   E3, E5   3

s7:  mov   D2, E6   2
     mul   E2, E6   3

s8:  mov   E3, D0   2
     add    5, D0   2
     mov   D0, E7   2
                   ---
                   40byte
```

【図１０】


第1の命令フォーマット(1)
第1の命令ﾌｨｰﾙﾄﾞ
第2の命令ﾌｨｰﾙﾄﾞ
第3の命令ﾌｨｰﾙﾄﾞ
第4の命令ﾌｨｰﾙﾄﾞ
第5の命令ﾌｨｰﾙﾄﾞ
第6の命令ﾌｨｰﾙﾄﾞ
第7の命令ﾌｨｰﾙﾄﾞ
基本命令語長
(1Byte)
(a)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
ﾚｼﾞｽﾀｱﾄﾞﾚｽ指定ﾌｨｰﾙﾄﾞ
(b)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
ﾚｼﾞｽﾀｱﾄﾞﾚｽ指定ﾌｨｰﾙﾄﾞ
付加情報ﾌｨｰﾙﾄﾞ
(c)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
ﾚｼﾞｽﾀｱﾄﾞﾚｽ指定ﾌｨｰﾙﾄﾞ
(d)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
ﾚｼﾞｽﾀｱﾄﾞﾚｽ指定ﾌｨｰﾙﾄﾞ
付加情報ﾌｨｰﾙﾄﾞ
(e)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
(f)
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
付加情報ﾌｨｰﾙﾄﾞ

【図１１】

第1の命令ﾌｫｰﾏｯﾄ(1)-(a)
ADD Dm,Dn　　：加算
CMP Dm,Dn　　：比較演算
CMP Am,An　　：比較演算
MOV(Am),Dn　　：ﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV Dm,(An)　　：ﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
MOV Dm,Dn　　：ﾚｼﾞｽﾀｰ→ﾚｼﾞｽﾀ間転送
MOV Am,An　　：ﾚｼﾞｽﾀｰ→ﾚｼﾞｽﾀ間転送
…

第1の命令ﾌｫｰﾏｯﾄ(1)-(b)
…

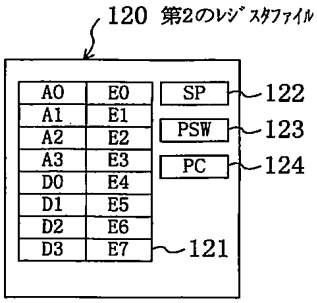
第1の命令ﾌｫｰﾏｯﾄ(1)-(c)
CLR Dm　　：ﾃﾞｰﾀ0ｸﾘｱ
INC Dm　　：ﾃﾞｰﾀ1ｲﾝｸﾘﾒﾝﾄ
INC Am　　：ﾃﾞｰﾀ1ｲﾝｸﾘﾒﾝﾄ
EXTB Dm　　：符号付きﾊﾞｲﾄﾃﾞｰﾀ→ﾜｰﾄﾞﾃﾞｰﾀ拡張
EXTUB Dm　　：符号無しﾊﾞｲﾄﾃﾞｰﾀ→ﾜｰﾄﾞﾃﾞｰﾀ拡張
EXTH Dm　　：符号付きﾊｰﾌﾜｰﾄﾞﾃﾞｰﾀ→ﾜｰﾄﾞﾃﾞｰﾀ拡張
EXTHU Dm　　：符号無しﾊｰﾌﾜｰﾄﾞﾃﾞｰﾀ→ﾜｰﾄﾞﾃﾞｰﾀ拡張
MOV SP,An　　：SP→ﾚｼﾞｽﾀ間転送
…

第1の命令ﾌｫｰﾏｯﾄ(1)-(d)
ADD imm8,An　　：8ﾋﾞｯﾄ即値加算
ADD imm8,Dn　　：8ﾋﾞｯﾄ即値加算
MOV imm16,An　　：16ﾋﾞｯﾄ即値転送
MOV imm16,Dn　　：16ﾋﾞｯﾄ即値転送
MOV(abs16),Dn　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOVBU(abs16),Dn　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
/ﾊﾞｲﾄﾃﾞｰﾀ符号拡張
MOVHU(abs16),Dn　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
/ﾜｰﾄﾞﾃﾞｰﾀ符号拡張
MOV Dm,(abs16)　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
MOVBU Dm,(abs16)　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
/ﾊﾞｲﾄﾃﾞｰﾀ符号拡張
MOVHU Dm(abs16)　　：16ﾋﾞｯﾄ絶対ｱﾄﾞﾚｽ指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
/ﾜｰﾄﾞﾃﾞｰﾀ符号拡張
MOV(disp8,SP),An　　：SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV(disp8,SP),Dn　　：SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV Am,(disp8,SP)　　：SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
MOV Dm,(disp8,SP)　　：SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
…

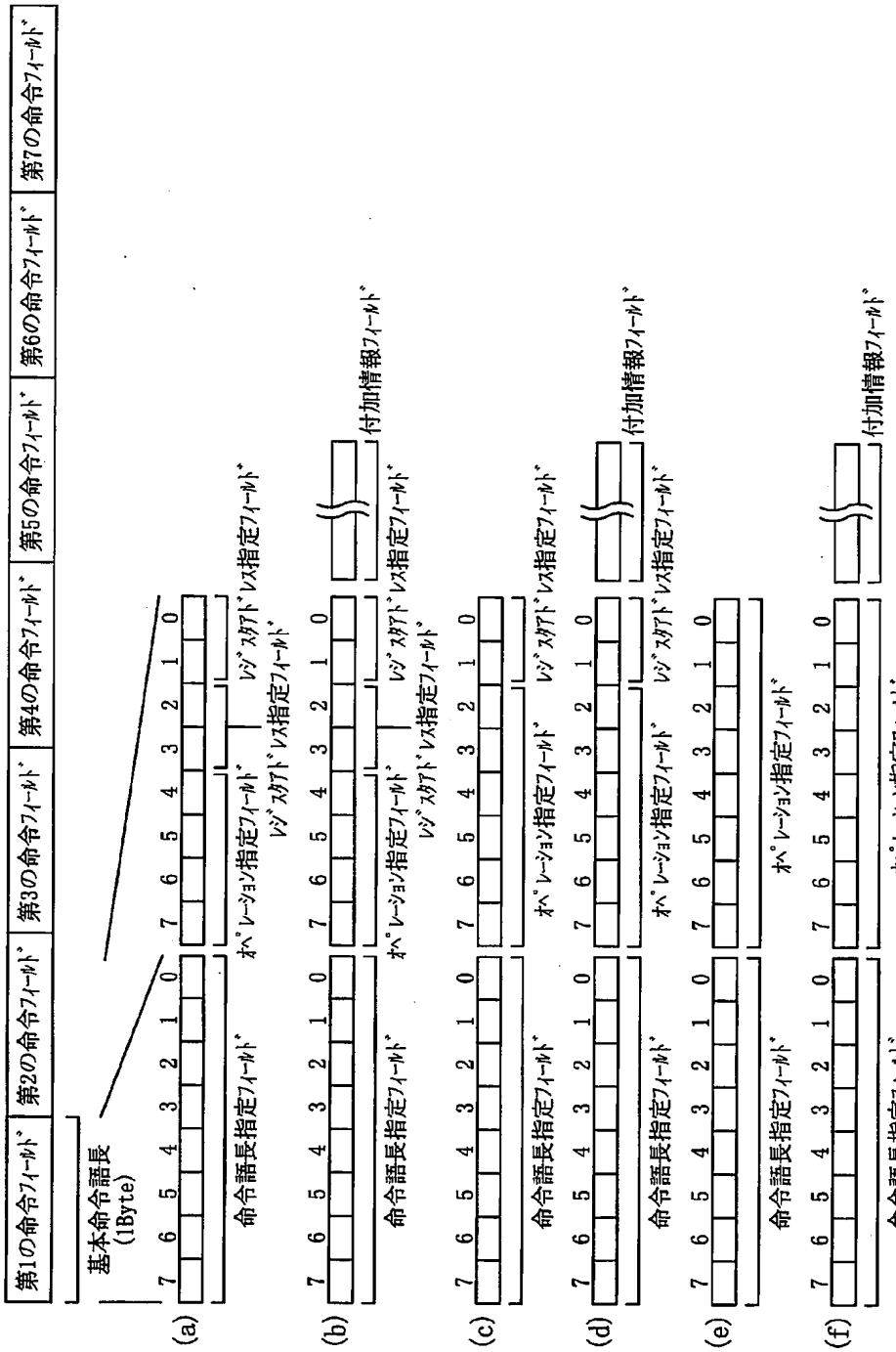
第1の命令ﾌｫｰﾏｯﾄ(1)-(e)
NOP　　：No Operation
LOOP #CC　　：ﾙｰﾌﾟ制御命令
…

第1の命令ﾌｫｰﾏｯﾄ(1)-(f)
BR #CC(disp8,PC)　　：PC相対条件分岐
JMP (disp8,PC)　　：PC相対絶対分岐
…

【図１８】



【図１２】


第1の命令フォーマット(2)
第1の命令ﾌｨｰﾙﾄﾞ
第2の命令ﾌｨｰﾙﾄﾞ
第3の命令ﾌｨｰﾙﾄﾞ
第4の命令ﾌｨｰﾙﾄﾞ
第5の命令ﾌｨｰﾙﾄﾞ
第6の命令ﾌｨｰﾙﾄﾞ
第7の命令ﾌｨｰﾙﾄﾞ
基本命令語長
(1Byte)
(a)
(b)
(c)
(d)
(e)
(f)
命令語長指定ﾌｨｰﾙﾄﾞ
ﾚｼﾞｽﾀｱﾄﾞﾚｽ指定ﾌｨｰﾙﾄﾞ
ｵﾍﾟﾚｰｼｮﾝ指定ﾌｨｰﾙﾄﾞ
付加情報ﾌｨｰﾙﾄﾞ

【図１３】

第1の命令ﾌｫｰﾏｯﾄ(2)－(a)
SUB Dm, Dn　　：減算
MOV(Am), An　　：ﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV Am, (An)　　：ﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
…

第1の命令ﾌｫｰﾏｯﾄ(2)－(b)
MOV(Ai, Dn), Dn　　：ｲﾝﾃﾞｯｸｽ付きﾚｼﾞｽﾀ間接ﾒﾓﾘ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
…

第1の命令ﾌｫｰﾏｯﾄ(2)－(c)
…

第1の命令ﾌｫｰﾏｯﾄ(2)－(d)
ADD imm16, An　　：16ﾋﾞｯﾄ即値加算
ADD imm16, Dn　　：16ﾋﾞｯﾄ即値加算
…

第1の命令ﾌｫｰﾏｯﾄ(2)－(e)
RTI　　：割り込み状態からの復帰
…

第1の命令ﾌｫｰﾏｯﾄ(2)－(f)
…

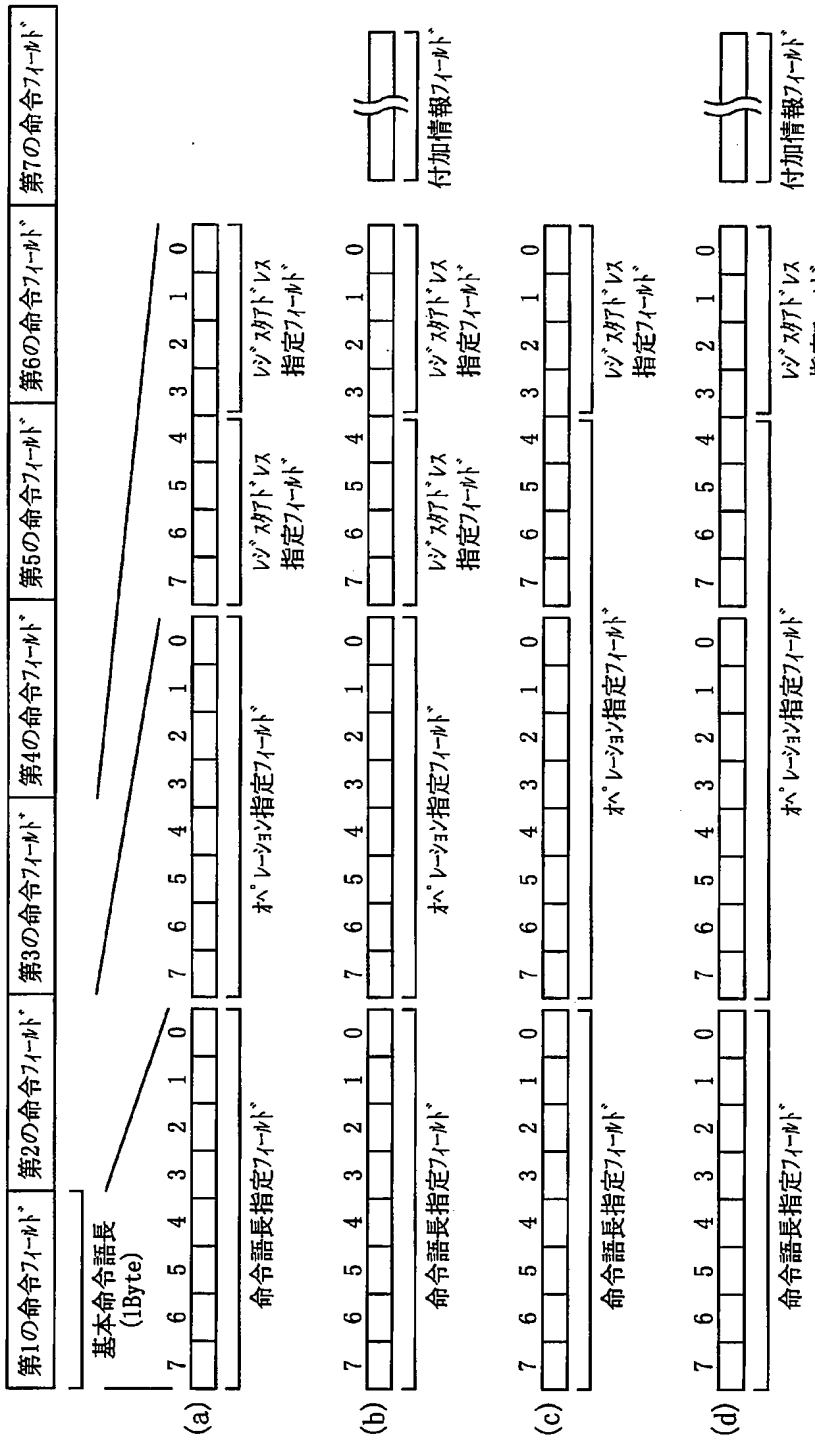
【図１９】

ﾚｼﾞｽﾀ名	命令ｺｰﾄﾞ上でのﾋﾞｯﾄ割り付け	物理的なﾚｼﾞｽﾀ番号	物理的なﾚｼﾞｽﾀ名
A0	00	1000	汎用ﾚｼﾞｽﾀ
A1	01	1001	汎用ﾚｼﾞｽﾀ
A2	10	1010	汎用ﾚｼﾞｽﾀ
A3	11	1011	汎用ﾚｼﾞｽﾀ
D0	00	1100	汎用ﾚｼﾞｽﾀ
D1	01	1101	汎用ﾚｼﾞｽﾀ
D2	10	1110	汎用ﾚｼﾞｽﾀ
D3	11	1111	汎用ﾚｼﾞｽﾀ
E0			
E1			
E2			
E3			
E4			
E5			
E6			
E7			

【図２０】

ﾚｼﾞｽﾀ名	命令ｺｰﾄﾞ上でのﾋﾞｯﾄ割り付け	物理的なﾚｼﾞｽﾀ番号	物理的なﾚｼﾞｽﾀ名
A0	1000	1000	汎用ﾚｼﾞｽﾀ
A1	1001	1001	汎用ﾚｼﾞｽﾀ
A2	1010	1010	汎用ﾚｼﾞｽﾀ
A3	1011	1011	汎用ﾚｼﾞｽﾀ
D0	1100	1100	汎用ﾚｼﾞｽﾀ
D1	1101	1101	汎用ﾚｼﾞｽﾀ
D2	1110	1110	汎用ﾚｼﾞｽﾀ
D3	1111	1111	汎用ﾚｼﾞｽﾀ
E0	0000	0000	汎用ﾚｼﾞｽﾀ
E1	0001	0001	汎用ﾚｼﾞｽﾀ
E2	0010	0010	汎用ﾚｼﾞｽﾀ
E3	0011	0011	汎用ﾚｼﾞｽﾀ
E4	0100	0100	汎用ﾚｼﾞｽﾀ
E5	0101	0101	汎用ﾚｼﾞｽﾀ
E6	0110	0110	汎用ﾚｼﾞｽﾀ
E7	0111	0111	汎用ﾚｼﾞｽﾀ

【図１６】


第2の命令フォーマット
第1の命令フィールド
第2の命令フィールド
第3の命令フィールド
第4の命令フィールド
第5の命令フィールド
第6の命令フィールド
第7の命令フィールド
基本命令語長
(1Byte)
7 6 5 4 3 2 1 0
(a)
命令語長指定フィールド
オペレーション指定フィールド
レジスタアドレス
指定フィールド
レジスタアドレス
指定フィールド
(b)
命令語長指定フィールド
オペレーション指定フィールド
レジスタアドレス
指定フィールド
付加情報フィールド
(c)
命令語長指定フィールド
オペレーション指定フィールド
レジスタアドレス
指定フィールド
(d)
命令語長指定フィールド
オペレーション指定フィールド
レジスタアドレス
指定フィールド
付加情報フィールド

【図１７】

第2の命令ﾌｫｰﾏｯﾄ(a)
```
ADD Rm,Rn          : 加算
SUB Rm,Rn          : 減算
CMP Rm,Rn          : 比較演算
MOV(Rm),Rn         : ﾒﾓﾘｰ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV Rm,(Rn)        : ﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
MOV Rm,Rn          : ﾚｼﾞｽﾀｰ→ﾚｼﾞｽﾀ間転送
...
```

第2の命令ﾌｫｰﾏｯﾄ(b)
```
ADD Rm,Rn,Rd       : 加算
SUB Rm,Rn,Rd       : 減算
MOV(Ri,Rm),Rn      : ｲﾝﾃﾞｯｸｽ付きﾚｼﾞｽﾀ間接ﾒﾓﾘｰ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
...
```

第2の命令ﾌｫｰﾏｯﾄ(c)
```
...
```

第2の命令ﾌｫｰﾏｯﾄ(d)
```
ADD imm 16,Rn      : 16ﾋﾞｯﾄ即値加算
ADD imm 16,Rn      : 16ﾋﾞｯﾄ即値加算
MOV(disp8,SP),Rn : SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾒﾓﾘｰ→ﾚｼﾞｽﾀ間転送(ﾛｰﾄﾞ)
MOV Rm,(disp8,SP): SP(ｽﾀｯｸﾎﾟｲﾝﾀ)相対指定によるﾚｼﾞｽﾀｰ→ﾒﾓﾘ間転送(ｽﾄｱ)
...
```